09/752,160

L Number	Hits	Search Text	DB	Time stamp
-	104221	(backup or back-up)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:52
-	696460	recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:52
-	2734	(monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:56
-	2798	detect$4 with harmful	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:57
-	1958	protect$4 same download$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:57
-	3751	((backup or back-up)) same recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:58
-	2	(((backup or back-up)) same recover$4) same ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:59
-	46	(((backup or back-up)) same recover$4) and ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 14:59
-	1	((((backup or back-up)) same recover$4) same ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)) and (detect$4 with harmful) and (protect$4 same download$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:00
-	1	((((backup or back-up)) same recover$4) same ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)) and (detect$4 with harmful)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:00

-	1	((((backup or back-up)) same recover$4) same ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)) and (protect$4 same download$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:00
-	3346	(714/?).ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	5	((((backup or back-up)) same recover$4) and ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)) and ((714/?).ccls.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:00
-	1934	(365/?).ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	2886	(710/?).ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	2206	(711/?).ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	657	(712/?).ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	10791	((714/?).ccls.) or ((365/?).ccls.) or ((710/?).ccls.) or ((711/?).ccls.) or ((712/?).ccls.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02
-	6	(((714/?).ccls.) or ((365/?).ccls.) or ((710/?).ccls.) or ((711/?).ccls.) or ((712/?).ccls.)) and ((((backup or back-up)) same recover$4) and ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:02

-	1	((((714/?).ccls.) or ((365/?).ccls.) or ((710/?).ccls.) or ((711/?).ccls.) or ((712/?).ccls.)) and ((((backup or back-up)) same recover$4) and ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$))) not (((((backup or back-up)) same recover$4) and ((monitor$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with message$)) and ((714/?).ccls.))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/22 15:03

09/759,160

L Number	Hits	Search Text	DB	Time stamp
1	52112	(monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:16
2	559	message$ adj download$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:00
3	10283	application$ adj layer$	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:03
4	134611	back adj up or back-up or backup	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:04
5	119199	harmfull or virus$2	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:13
6	642	(harmfull or virus$2) adj3 data	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:13
7	0	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) same (back adj up or back-up or backup) same ((harmfull or virus$2) adj3 data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:16
8	5	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (back adj up or back-up or backup) and ((harmfull or virus$2) adj3 data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:19
9	1	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (message$ adj download$4) and (application$ adj layer$) and (back adj up or back-up or backup)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:37
10	696460	recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:20

11	1955	automatic$4 adj protect$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
12	30	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
14	1	((((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)) and recover$4) and (automatic$4 adj protect$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
13	16	(((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)) and recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
15	0	(message$ adj download$4) with (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
16	30	(message$ adj download$4) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
17	1	(message$ adj download$4) same (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
18	4	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (message$ adj download$4) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:40
19	2694	(back adj up or back-up or backup) with recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
20	4	((back adj up or back-up or backup) with recover$4) and ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:42

21	12	((back adj up or back-up or backup) with recover$4) same ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:46
22	2	((back adj up or back-up or backup) with recover$4) same (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:46
23	20	((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
24	1	(((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)) and (message$ adj download$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
25	3	(((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)) and (harmfull or virus$2)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:55
26	996	(back adj up or back-up or backup) adj3 recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:55
27	4	((back adj up or back-up or backup) adj3 recover$4) with ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:56
28	104	((back adj up or back-up or backup) adj3 recover$4) with (computer)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:57
29	60	26.ti.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:57

09/750,160

L Number	Hits	Search Text	DB	Time stamp
1	52112	(monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:16
2	559	message$ adj download$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:00
3	10283	application$ adj layer$	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:03
4	134611	back adj up or back-up or backup	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:04
5	119199	harmfull or virus$2	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:13
6	642	(harmfull or virus$2) adj3 data	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:13
7	0	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) same (back adj up or back-up or backup) same ((harmfull or virus$2) adj3 data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:16
8	5	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (back adj up or back-up or backup) and ((harmfull or virus$2) adj3 data)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:19
9	1	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (message$ adj download$4) and (application$ adj layer$) and (back adj up or back-up or backup)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:37
10	696460	recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:20

11	1955	automatic$4 adj protect$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
12	30	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
14	1	((((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)) and recover$4) and (automatic$4 adj protect$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:24
13	16	(((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$) and (back adj up or back-up or backup)) and recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
15	0	(message$ adj download$4) with (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
16	30	(message$ adj download$4) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
17	1	(message$ adj download$4) same (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:36
18	4	((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (message$ adj download$4) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:40
19	2694	(back adj up or back-up or backup) with recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
20	4	((back adj up or back-up or backup) with recover$4) and ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data)) and (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:42

21	12	((back adj up or back-up or backup) with recover$4) same ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:46
22	2	((back adj up or back-up or backup) with recover$4) same (application$ adj layer$)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:46
23	20	((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
24	1	(((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)) and (message$ adj download$4)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:51
25	3	(((back adj up or back-up or backup) with recover$4) and (automatic$4 adj protect$4)) and (harmfull or virus$2)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:55
26	996	(back adj up or back-up or backup) adj3 recover$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:55
27	4	((back adj up or back-up or backup) adj3 recover$4) with ((monitor$4 or detect$4 or track$4 or check$4) with (predeterm$4 or preset$4 or predefin$4) with (message$ or data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/07/24 17:55

US-PAT-NO: 6502102

DOCUMENT-IDENTIFIER: US 6502102 B1
See image for Certificate of Correction

TITLE: System, method and article of manufacture for a table-driven automated scripting architecture

---------- KWIC ---------

Detailed Description Text - DETX (449):
The incremental value of the daily work performed on the development project is high. This investment must be protected from problems arising from hardware and software failure, and from erroneous user actions and catastrophes such as fires or floods. The repositories and other development information must therefore be backed up regularly. **Backup** and restore procedures and tools must be tested to ensure that system components can be **recovered** as anticipated. The large volumes of complex data generally require automation of backups and restores.

Detailed Description Text - DETX (684):
Firewalls are often thought of as THE answer to network security. There is a common misconception that purchasing and installing the "best" firewall available may **automatically protect** your network from the Internet. This is not necessarily true. In fact there are many factors to consider when choosing a firewall, and when placing and configuring that firewall in your environment. First of all, consider the type of network connection your are trying to protect. Firewalls are not only used to separate your intranet from the Internet, they can also be used to segregate a particularly sensitive or particularly insecure area of your intranet from the rest of your network. Depending on the services one wants to provide your users and what risk one is willing to accept, your choice of the "best" firewall implementation may change.

Detailed Description Text - DETX (692):
Backup and Recovery

Detailed Description Text - DETX (693):
People kick over servers, accidentally delete files, and spill coffee on machines. For these reasons and a host of others, Net Centric resources must be backed up in a manner so that they can be recovered. This does not mean dumping a bunch of files onto data tapes and stacking them in a comer of the server room. An effective **backup and recovery** strategy should address how **backups** may be taken, the media on which they may be stored, the location where they may be stored, and the frequency with which they may be taken. **Backups** should also be periodically tested to make sure that they are **recoverable,** for example to make sure the **backup** tape drive is still working. When designing your backup strategy one should also consider the specific types of applications, databases, and hardware which are in use in your environment. For example an Oracle database may probably not be recoverable from a .tar file. In addition to software resources, consider what would happen if your router or your ISP link were to go down. It may be necessary to maintain a backup link to a secondary service provider in the event that your ISP goes down for an extended period of time.

Detailed Description Text - DETX (1061):
Pre-Installation Suggestions Do not install Site Server on a Backup Domain Controller. Do not install Exchange Server on a Site Server. Both products are resource intensive. Do not install Proxy Server on a Site Server. Do not install Site Server on a Clustered NT System (MSCS). One can install Site Server onto a Windows Load Balancing Service (WLBS). Remove Content Analyzer from Visual Studio. Only install Site Server on a NTFS Drive. Disable or Remove all Anti **Virus** software during entire install process. Do not change ANY setting in IIS before installing Site Server (On a clean/new install). Have at least one gig free of disk space. Verify that virtual memory is set to at least 128 MB during the install process. Give your account

administrative
privileges on the local machine.

Detailed Description Paragraph Table - DETL (49):
Title Description & Responsibilities Technical Manager Typically an IS
department head with responsibility for the purchase and/or support of hardware and software. In configuration management, this role is more software oriented. Other responsibilities include: Assign development and
support staff to projects. Review (accept/reject) technical approach proposed
for projects. Monitor development and support budgets and personnel - status
of projects. Network System This individual is responsible for the installation, Administrator maintenance and support of the Unix and Windows
NT servers including operating system, file systems, and applications. Other
responsibilities include: Operating system installation, patch upgrades,
migrations and compatibility with other applications. Installation and support
of proper backup/restore systems. Installation and support of other peripherals required for installed (or to be installed) applications. Proper
portion of the present description of hardware configuration and setup.
Maintenance of Windows Domain users and Groups as well as other security
issues. Database The DBA is responsible for proper creation and Administrator
maintenance of production and system test databases. The integrity of the
database, as well as **recovery** using **backup**/restore and logging, are priorities for the DBA. Other responsibilities include: Assist developers in
maintaining development databases by automating **backup/recovery,** applying
changes to database schema, etc. Provide support for tuning, sizing and
locating database objects within allocated database space. Applying change
requests to databases. Ideally maintain entity relationship diagrams for
databases. Maintenance of database users and other database-related security
issues Source Code Individual responsible for development and Librarian
maintenance of source code control tools, training materials, and storage
areas. The Source Code Librarian is also responsible for the integrity of the
source code environment. Additionally: Establishes source code directories for
new projects. Provides reports on source code environment status and usage

per project. Provides assistance/information as needed regarding objects to
check out for system test. Assists production operations in building/moving
all applications into production. Business Analyst Individual or individuals
responsible for managing the detailed design, programming, and unit testing
of application software. Other responsibilities include: Developing/reviewing
detailed designs. Developing/reviewing unit test plans, data, scripts, and
output. Managing application developers. Application Individual or
individuals responsible for making Developer changes to source code defined by
management. This person typically: Checks source code out of the source code
environment. Modifies code per user requirements or other development portion
of the present description. Unit tests modifications in the development
environment. Checks modified code back into source code environment in
preparation for system test. System Tester This person or team is directly
responsible for Integration Tester system testing or integration testing of an
application prior to implementing in production. This may also take the form
of performance testing. Typically, a system or integration test person or
team may be responsible for: Following production operation procedures for
installing a new application in the appropriate test environment. Develop and
execute a test plan to properly exercise new application including new,
modified, and unmodified functionality. Reporting results of test. Vendor
For the purposes of this portion of the present description, a vendor is
defined as an organization from which software has been purchased for use by
the clients systems. Alternatively, a vendor may distribute final installable
media in the form of tape or CD with upgrades or new release of application.
A vendor may: Make modifications to application code at vendor offices or
within the engagement development environment. Provide necessary information
to Source Code Librarian to store new code. Assist Source Code Librarian in
transferring modifications to the engagement system test environment.
Participate in system test (or performance test).

Detailed Description Paragraph Table - DETL (61):

Step Step Description Notes 1 Install "Oracle 8 Enterprise Edition"
(Version 8.0.3.0.0 for Windows Though these NT) steps describe Run Setup on
the installation CD. the installation Choose the installation language, then
select OK. on a Windows Choose the Company name, and change the default
install directory to NT platform, C:.backslash.Oracle, then select OK. they
are nearly Select Yes when asked whether to have the installation program
make identical to the changes to the PATH variable installation Choose to
install Oracle 8 Enterprise Edition. process on the Select where the Oracle
portion of the present description should be UNIX installed. The default is
to leave it on the CD. platform. 2 Create a directory for the application
database. Start the windows explorer Select the directory where Oracle is
installed (C:.backslash.Oracle) then the subdirectory Database Create a new
folder for the Database files. Ex.
"C:.backslash.Oracle.backslash.ReTA" 3 At
this point a full operating system backup should be made, and the backup set
stored. In future, if the database server goes down, this backup may be used
to quickly restore the server to a point where the Oracle **Recovery** Manager can
take over and complete the **backup.** 4 Add registry keys for the database. The
key locations are This key
HKEY_LOCAL_MACHINE.backslash.SOFTWARE.backslash.Oracle.backslash..
identifies
the Use the Start Menu to run the regedit application active database Browse
to the above key. to Oracle on Right click on the entry ORACLE_SID and select
Modify. startup. Set the key value to RETA (or the SID of the Database if this
has been modified. Create a new key, NLS_DATE_FORMAT, and set the value to
"DD- MM-YY HH24:MM:SS" (include the quotation marks) 5 Perform the initial
database creation. This batch file Run the batch file Create ReTA
Database.bat located in the is expects Database.backslash.CreateDB
subdirectory of the Architecture directory of the RETARUN.sql supplied media.
and NOTE: The following batch files and database scripts may sometimes
RETA1RUN.sql generate errors of the form "Table / View does not exist." This
is to be located because the scripts delete before trying to create objects -
if the scripts in the same are being run for the first time these objects may

not exist and the directory. errors may be generated. This is not a cause for
concern. 6 Register your new databases with the TNS listener service to enable
This step other computers on the network to see it. enables Open the file
listener.ora located in the Net80.backslash.admin directory of the Oracle8
Client Oracle directory. communication Create entries identical to the ORCL
entry at the end of the file, with with the data the SIDs replaced by PROS,
or the SID created in step 4. server. Note: copy the entire code block - i.e.
four lines of code. The inserted code is the following: (SID_DESC = ReTA
Development Database) (GLOBAL_DBNAME = <Your computer name here>)
(SID_NAME = <Your database SID here>)) Stop and restart the service
Oracle TNS Listener 7 Create local connections to the new database. This step
Use the start menu to run the program Oracle for Win NT / Net8 Easy provides
access Config. to the database Note: If one gets a Dr. Watson error on
Java.exe, set the display to 256 from colors. SQL*Plus, Select Add New
Service, and supply a service name e.g. "RETA1" Oracle Select Bequeath (local
database). Navigator or Select Next. other Oracle Enter the database SID used
in the database creation script (RETA by administrative default) tools.
Select Test Service (Username: system; Password: Manager) and when the test is
successful push Done Select Next, then Finish.

US-PAT-NO: 5513314

DOCUMENT-IDENTIFIER: US 5513314 A

TITLE: Fault tolerant NFS server system and mirroring protocol

---------- KWIC ---------

Detailed Description Text - DETX (4):
In accordance with the preferred embodiments of the present invention, a fault tolerant protocol is implemented for a specific class of remote procedure calls (RPCs) transferred via the LAN 16 as a series of one or more datagrams. Specifically, the class of RPCs encompassed by the fault tolerant protocol include those known as Network Filesystem (NFS) requests. In general, NFS requests provide for two categories of operations: inquiries and updates. Inquiry requests include read data, get attributes, look up, read directory, read link, status, and null. Update requests include write data, set attributes, rename, remove directory, remove file, link, create file, make directory, and make symbolic link. These NFS requests are **monitored** and managed by the fault tolerant protocol of the present invention in a manner that results in the mirroring of all **data** within **predetermined** filesystems present on a primary 12 and least one secondary 14 file server. The mirroring of data to both the primary and secondary file servers 12, 14 is performed essentially concurrently in response to any client workstation 18, 20 that issues NFS requests with respect to the mirrored filesystems.

Detailed Description Text - DETX (13):
On both the primary files server 12 and secondary file server 14, the datagram representing the NFS write request is processed by a substantially conventional TCP/IP stack. In relevant part, this network stack includes a physical layer, a data link layer, a network layer, a transport layer, a session layer and an. **application layer**.

Detailed Description Text - DETX (16):

Finally, the **application layer** provides for well-known file services, such as file transfer and remote file access. An NFS server layer is the preferred embodiment of the **application layer** used by the present invention. Each read, write or other NFS request is managed through the NFS server under the control of generally respective network control processes (conventionally nfsd processes).

Detailed Description Text - DETX (33):

In each of these events, the primary server 12 is left sleeping on the DRC entry for an acknowledgment datagram that is not received. However, in accordance with the present-invention, a sleep timer is set by the primary server 12 in putting the nfsd process to sleep on DRC entry. The nfsd process awakes 86 on timeout of the sleep timer in the absence of any received acknowledge datagram. Alternately, the sleep timer is effectively expired upon the aging of the DRC entry through operation of the DRC-LRU algorithm. In either event, the primary server 12 then transitions to a **backup** failure **recovery** mode 88.

US-PAT-NO: 5913219

DOCUMENT-IDENTIFIER: US 5913219 A

TITLE: Database recovery apparatus and method of using dual plane nonvolatile memory

---------- KWIC ---------

Brief Summary Text - BSTX (19):

A database **recovery** apparatus using a dual plane nonvolatile memory according to a first embodiment of the present invention to accomplish the above described object is characterized in that it comprises a memory for storing a database processing program; a power **monitor** and control circuit which receives a **backup/recovery** state signal, supplies a power source to maintain the **backup** state for a **predetermined** time if a power failure occurs in the **backup** state, and prevents the **backup** state from being performed by outputting an interrupt signal if the power failure occurs in a case other than the **backup** state; a dual plane nonvolatile memory comprising a volatile memory and nonvolatile memory divided into a plurality of block units whereby a chip is selected by a chip select signal, and performing a **recovery** process which copies block **data** of the nonvolatile memory to the volatile memory block in response to a **backup/recovery** control signal and an address read/write control signal and a **backup** process which overwrites the **data** of volatile memory block to the nonvolatile memory block; and a database processing circuit for loading the program from the memory, outputting the **backup/recovery** control signal to the power **monitor** and control circuit, outputting a chip select signal to the dual plane nonvolatile memory, and outputting the address and the **backup/recovery** control signal so that the **recovery** process is performed, and the **data** of the volatile memory block of the dual plane nonvolatile memory is updated through the address and the read/write control, and thereafter the **backup** process is performed by outputting the address and the **backup recovery** control signal.

Brief Summary Text - BSTX (21):

A database **recovery** apparatus using a dual plane nonvolatile memory according to a second embodiment of the present invention to accomplish the above described object is characterized in that it comprises a memory for storing a database processing program; a power **monitor** and control circuit which receives a **backup/recovery** state signal, supplies a power source to maintain the **backup** state for a **predetermined** time if a power failure occurs in the **backup** state, and prevents the **backup** state from being performed by outputting an interrupt signal if the power failure occurs in a case other than the **backup** state; a first dual plane nonvolatile memory and a second dual plane nonvolatile memory comprising a volatile memory and nonvolatile memory divided into a plurality of block units whereby a chip is selected by a chip select signal, and performing a **recovery** process which copies block **data** of the nonvolatile memory to the volatile memory block in response to a **backup/recovery** control signal and an address read/write control signal and a **backup** process which overwrites the **data** of the volatile memory block to the nonvolatile memory block; and a database processing circuit for loading the program from the memory, outputting the **backup/recovery** control signal to the power **monitor** and control circuit, outputting a chip select signal to the first and second dual plane nonvolatile memory, and outputting the address and the **backup/recovery** control signal so that the **recovery** process is performed, the **data** of the volatile memory block of the first and second dual plane nonvolatile memory is updated through the address and the read/write control, and thereafter the address and the **backup recovery** control signal are outputted to the first dual plane nonvolatile memory, and then if the **backup** process fails, the database of the first dual plane nonvolatile memory is updated by reading the **data** prior to update of the second dual plane nonvolatile memory, and if the **backup** process of the first dual plane nonvolatile memory succeeds, the address and the **backup/recovery** control signal are outputted to the second dual plane nonvolatile memory and then if the **backup** process fails, the database of the second dual plane nonvolatile memory is updated by

reading the
updated **data** of the first dual plane nonvolatile memory.

US-PAT-NO: 6507562

DOCUMENT-IDENTIFIER: US 6507562 B1

TITLE: DYNAMIC OPTIMIZATION FOR RECEIVERS USING DISTANCE BETWEEN A REPAIR HEAD AND A MEMBER STATION IN A REPAIR GROUP FOR RECEIVERS HAVING A CLOSELY KNIT TOPOLOGICAL ARRANGEMENT TO LOCATE REPAIR HEADS NEAR THE MEMBER STATIONS WHICH THEY SERVE IN TREE BASED REPAIR IN RELIABLE MULTICAST PROTOCOL

---------- KWIC ---------

Detailed Description Text - DETX (34):
If this timer expires, it indicates that the sender has paused and allows members to report and **recover** any lost packets without having to wait for the sender to start sending new data.

Detailed Description Text - DETX (58):
Each repair head monitors the operation of the members of its respective repair group to ensure that the members are functioning properly. Likewise, each of the members of a given repair group monitor the operation of the repair head associated with that group to ensure proper functioning of the head. If a repair head determines that a member of its group is no longer functioning (e.g., as a result of failure of the member to acknowledge receipt of special **monitoring messages** after a **predetermined** number of **messages** have been transmitted and/or a **predetermined** time period for response has elapsed), the repair head may prune that member from its group.

Detailed Description Text - DETX (79):
When a member receives the beacon packet, it immediately sends an acknowledgment to its repair head indicating whether it has received all of the packets transmitted, or requires packet retransmission. If the beacon from the sender is received, but a member has not acknowledged receipt of all data packets, a monitoring message is transmitted from the repair head

associated
with that member. If the member does not acknowledge receipt of such message
to the repair head sending the monitoring message, the repair head may retransmit the monitoring message. If, after a **predetermined** number of retransmissions of the **monitoring message,** the member has still failed to
acknowledge receipt, the repair head prunes the member from the tree. When all
members have either acknowledged receipt of all data packets to the repair head
or have been pruned from the tree, the repair head terminates its session.

Detailed Description Text - DETX (198):
The following are some of the tree optimization techniques: When a member
hears a Hello from a different repair head in the region that is closer than
its current repair head, the member attempts to re-affiliate to the closer
repair head. Note that a repair head has to perform loop avoidance checks
before choosing to re-affiliate with the closer repair head. Without loop
avoidance checks, improper tree formation (or tree disintegration) may result
when a repair head chooses to affiliate with a repair head that is at or below
its level/depth in the tree hierarchy. When two repair heads are found to be
close to each other, one of the repair heads can volunteer to resign in favor
of the other repair head. Typically the repair head that is better suited can
continue to be a repair head while the other repair head can resign. In
situations where both the repair heads are found to be suitable, tie breaker
techniques such as the repair head that has fewer members, or the repair head
that has the lowest unicast address and port number combination, can be used to
resolve the condition. To ensure quick and smooth re-affiliations of its
members, a resigning repair head can include the details of any **backup** repair
head (network address, unicast port number, worst case TTL distance from the
backup repair head to the members) in the Hello message. The **backup** repair
head details are informatory in nature and members with better alternatives can
choose to ignore this information and re-affiliate with a different repair
head. The details of the **backup** repair head help the members that do not hear

Hellos from any other repair head other than the currently affiliated repair
head. The repair head can adopt a strategy wherein the members that are
considered to be farthest are repaired using unicast and those that are
considered closer are repaired using multicast. When this strategy is in use,
the repair heads can be limited to accept only a few unicast members.

Detailed Description Text - DETX (211):
This timer is canceled if an ACK is sent using the triggering mechanism
described above. If this timer expires, it indicates that the sender has
paused and allows members to report and **recover** any lost packets without having
to wait for the sender to start sending new data.

Detailed Description Text - DETX (278):
Receivers joining the multicast group after data transmission has startedhave two options for **recovering** data previously sent:

Detailed Description Text - DETX (279):
Recover as much data previously sent as possible. This option allows the
receiver to request retransmission of all the previously sent data that its
repair head has cached. A repair head typically has at least the last 50
packets sent, in its cache.

Detailed Description Text - DETX (280):
Don't **recover** anything sent before the receiver joined. This option doesn't
attempt to **recover** any previously sent packets. The first data packet received
after the new member joins the repair tree is handed up to the application.
All previously sent packets are ignored.

Detailed Description Text - DETX (299):
The Internet architecture is represented by four layers which are termed, in
ascending interfacing order, the network interface, internetwork, transport and
application layers. These layers are arranged to form a protocol stack in each
communicating station of the network.

Detailed Description Text - DETX (301):
In general, the lower layers of the communications stack provide Internetworking services and the upper layers, which are the users of

these
services, collectively provide common network application services. The
application layer 12,112 provides services suitable for the different types of
applications using the internetwork, while the lower network interface layer
12,120 accepts industry standards defining a flexible network architecture
oriented to the implementation of local area networks (LANs).

Detailed Description Text - DETX (305):
Data transmission over the internetwork 12,100 therefore consists of generating data in, e.g., sending process 12,104 executing on the source
station 12,110, passing that data to the **application layer** 12,112 and down
through the layers of the protocol stack 12,125, where the data are sequentially formatted as a frame for delivery onto the medium 12,180 as bits.
Those frame bits are then transmitted over an established connection of medium
12,180 to the protocol stack 12,175 of the destination station 12,150 where
they are passed up that stack to a receiving process 12,174. Data flow is
schematically illustrated by solid arrows.

Detailed Description Text - DETX (309):
One approach to providing scalable reliable multicasting is to organize the
receivers into a tree structure so that each internal "node" of the tree is
responsible for helping its subordinates **recover** any lost packets and communicating status back to the sender. Many conventional algorithms exist
for constructing such a tree. For example, reliable multicast protocols such
as TMTP and RMTP build trees that are used for an entire data transfer session
without optimization. Lorax describes methods for generally enforcing member
limits. After such a tree is constructed, it may be further optimized as
network conditions change. The present invention is directed, in one aspect,
to defining characteristics of optimal trees and mechanisms for obtaining such
trees.

Detailed Description Text - DETX (314):
The present invention generally relates to a scalable, reliable multicast
transport protocol (TRAM) that supports bulk data transfer with a single sender

and multiple receivers of a computer internetwork, such as an intranet or Internet. In one aspect of the invention, TRAM uses reliable multicast repair trees that are optimized to implement local error **recovery** and to scale to a large number of receivers without substantially impacting the sender.

Detailed Description Text - DETX (319):
The invention provides many features, for example the features of the invention include: reliable multicast; single source to many receivers; scalable--ability to support a large receiver community; support local repair; support adaptive congestion control mechanisms to prevent network flooding; ordered data delivery; support unidirectional and multidirectional multicast environments during the initial building of the tree and for late joins, and reaffiliation during data transfer; control bandwidth used by multicast control messages during tree formation anti data transfer; scalable up to a million receivers; late joins without data **recovery**; support for real-time data and resilient category of applications; and, unordered data delivery.

Detailed Description Text - DETX (328):
Process of sensing congestion and **recovering** from it rather than aggravating it. The congestion control mechanism is rate based and is adaptive which enables the sender to sense and adjust to the rate at which the receivers can accept the data.

Detailed Description Text - DETX (396):
Further, since the Hello message is a multicast message, the Hello message can serve as a means to inform other RxGroup-heads and RxGroup-members in the neighborhood of its existence. This can be used to detect and optimize the number of heads in the neighborhood and can also serve to provide **backup** head information to other members in the neighborhood. The TTL scope in use field in the Hello message can be maintained by the non dependent members as a **backup** TTL, and can quickly re-affiliate upon losing its dependent RxGroup-head.

Detailed Description Text - DETX (424):

Members of trees that use MTHA for late joins still have a possible way to determine standby heads. Every RxGroup-member, while successfully affiliated to a RxGroup-head, processes the nearby RxGroup-head's multicast Hello messages and maintains a maximum **backup** TTL value that it may have to use reach one of these neighboring RxGroup-heads. The TTL values from Hello messages that indicate that the HSTATE as Not_Accepting_Members are ignored.

Detailed Description Text - DETX (425):
If the watchdog timer tracking the RxGroup-head expires N_HELLO_MISSES times, the RxGroup-member starts the re-affiliation process by sending a MS message with TTL scope of the message set to the **backup** head TTL scope maintained (if any, otherwise with expanding scope as in a late join). If the MS sent to the computed TTL scope does not yield HA messages, ERS mechanism will be pursued.

Detailed Description Text - DETX (447):
This timer is canceled if an ACK is sent using the triggering mechanism described above. If this timer expires, it indicates that the sender has paused and allows members to report and **recover** any lost packets without having to wait for the sender to start sending new data.

Detailed Description Text - DETX (471):
Recover as much previously sent data as possible. This option allows the receiver to ask for retransmissions of all the previously sent data that its repair head has cached. A repair head typically has at least the last 50 packets sent in its cache.

Detailed Description Text - DETX (472):
Do not **recover** anything sent before the receiver joined. This option doesn't attempt to **recover** any previously sent packets. The first data packet received after the new member joins the repair tree is handed up to the application. All previously sent packets are ignored.

Detailed Description Text - DETX (473):
Both of the above options require that the receiver join the multicast

repair tree before any data is given to the application. The method setLateJoinPreference is used to select one of the options listed above. Valid
arguments to this call are: LATE_JOIN_WITH_LIMITED_**RECOVERY**
LATE_JOIN_WITH_NO_**RECOVERY**

Detailed Description Text - DETX (484):
The data message is encapsulated in a TRAM header message and is sent to the
multicast group. The TRAM header among other things, include a sequence number
which enable the receiver TRAMs to order (if required) and detect packet loss.
After transmission, the message is moved to the Retrans-Q. The RxGroup-members
use a window mechanism to acknowledge the receipt of the multicast messages.
The message on the RetransQ undergoes the state transition (described earlier)
before being freed. If data cache usage is found to be above the high water
mark, then the congestion control and analysis operation on the RetransQ is
initiated to isolate and **recover** from the condition.

US-PAT-NO: 6115393

DOCUMENT-IDENTIFIER: US 6115393 A

TITLE: Network monitoring

---------- KWIC ---------

Detailed Description Text - DETX (16):
Management Workstation 12 is the operator interface. It collects and presents troubleshooting and performance information to the user. It is based on the SunNet Manager (SNM) product and provides a graphical network-map-based interface and sophisticated data presentation and analysis tools. It receives information from Monitor 10, stores it and displays the information in various ways. It also instructs Monitor 10 to perform certain actions. Monitor 10, in turn, sends responses and alarms to Management Workstation 12 over either the primary LAN or a **backup** serial link 14 using SNMP with the MIB extensions defined later.

Detailed Description Text - DETX (21):
For purposes of the present description, the Open Systems Interconnection (OSI) model will be presented as representative of structured protocol architectures. The OSI model, developed by the International Organization for Standardization, includes seven layers. As indicated in FIG. 2, there is a physical layer, a data link layer (DLL), a network layer, a transport layer, a session layer, a presentation layer and an **application layer,** in that order. As background for what is to follow, the function of each of these layers will be briefly described.

Detailed Description Text - DETX (22):
The physical layer provides the physical medium for the data transmission. It specifies the electrical and mechanical interfaces of the network and deals with bit level detail. The data link layer is responsible for ensuring an error-free physical link between the communicating nodes. It is

responsible
for creating and recognizing frame boundaries (i.e., the boundaries of the
packets of data that are sent over the network.) The network layer determines
how packets are routed within the network. The transport layer accepts data
from the layer above it (i.e., the session layer), breaks the packets up into
smaller units, if required, and passes these to the network layer for
transmission over the network. It may insure that the smaller pieces all
arrive properly at the other end. The session layer is the user's interface
into the network. The user must interface with the session layer in order to
negotiate a connection with a process in another machine. The presentation
layer provides code conversion and data reformatting for the user's
application. Finally, the **application layer** selects the overall network
service for the user's application.

Detailed Description Text - DETX (33):
Stated another way, a dialog is the exchange of messages and the associated
meaning and state that is inherent in any particular exchange at any layer. It
refers to the exchange between the peer entities (hardware or software) in any
communication. In those situations where there is a layering of protocols, any
particular message exchange could be viewed as belonging to multiple dialogs.
For example, in FIG. 4 Nodes A and B are exchanging packets and are engaged in
multiple dialogs. Layer 1 in Node A has a dialog with Layer 1 in Node B. For
this example, one could state that this is the data link layer and the nature
of the dialog deals with the message length, number of messages, errors and
perhaps the guarantee of the delivery. Simultaneously, Layer n of Node A is
having a dialog with Layer n of node B. For the sake of the example, one could
state that this is an **application layer** dialog which deals with virtual
terminal connections and response rates. One can also assume that all of the
other layers (2 through n-1) are also having simultaneous dialogs.

Detailed Description Text - DETX (70):
Second, MTM 34 is responsible for the delivery and reception of data to and
from the Management Workstation using the protocol appropriate to the network.

Primary and **backup** communication paths are provided transparently to the rest of the monitor modules (e.g. LAN and dial up link). It is capable of full duplex delivery of messages between the console and monitoring module. The messages carry event, configuration, test and statistics data.

Detailed Description Text - DETX (168):

DLL dialog statistics data structure 178, illustrated by FIG. 7c, includes several additional fields of information which only appear in these structures for dialogs for which state information can be kept (e.g. TCP connection). The additional fields identify the transport protocol (e.g., TCP) (field 184) and the application which is running on top of that protocol (field 186). They also include the identity of the initiator of the connection (field 188), the state of the connection (field 190) and the reason that the connection was closed, when it is closed (field 192). Finally, they also include a state.sub.-- pointer (field 194) which points to a history data structure that will be described in greater detail later. Suffice it to say, that the history data structure contains a short history of events and states for each end of the dialog. The state machine uses the information contained in the history data structure to loosely determine what the state of each of the end nodes is throughout the course of the connection. The qualifier "loosely" is used because the state machine does not closely shadow the state of the connection and thus is capable of **recovering** from loss of state due to lost packets or missed communications.

Detailed Description Text - DETX (231):

Though the Network Monitor operates in a promiscuous mode, it may occasionally fail to detect or it may, due to overload, lose a packet within a communication. If this occurs the state machine may not be able to accurately determine the state of the connection upon receipt of the next event. The problem is evidenced by the fact that the next event is not what was expected. When this occurs, the state machine tries to **recover** state by relying on state history information stored in the history table in field 212 to deduce what the

state is. To deduce the current state from historical information, the state
machine uses one of the two previously mentioned routines, namely, Look.sub.--
for.sub.-- Data.sub.-- State and Look.sub.-- at.sub.-- History.

Detailed Description Text - DETX (352):

Referring the FIG. 24, the details of the training procedure for adaptively
setting the Network Monitor thresholds are as follows. To begin training, the
Workstation sends a start learning command to the Network Monitors from which
performance data is desired (step 302). The start learning command disables
the thresholds within the Network **Monitor** and causes the Network **Monitor** to
periodically send **data for a predefined** set of network parameters to the
Management Workstation. (Disabling the thresholds, however, is not necessary.
One could have the learning mode operational in parallel with monitoring using
existing thresholds.) The set of parameters may be any or all of the previously
mentioned parameters for which thresholds are or may be defined.

US-PAT-NO: 6065073

DOCUMENT-IDENTIFIER: US 6065073 A
See image for Certificate of Correction

TITLE: Auto-polling unit for interrupt generation in a network
interface device

---------- KWIC ---------

Brief Summary Text - BSTX (6):
The advantages of LANs are numerous. By providing easy access to shared data (on server computer 14, for example), computer users are allowed to interpolate more effectively. Users are also able to share expensive peripheral devices such as printers, faxes and CD-ROMs between client computers 16. These peripheral devices are also coupled to the various client computers via LAN hardware 12. The cost of client computers may also be decreased by lessening the needs for high-capacity disk drives on individual workstations. By storing data on one or more central servers accessible through the LAN, this also provides an easier solution for **backup** of vital data.

Brief Summary Text - BSTX (12):
Layer 7, the **application layer,** is responsible for specialized network functions such as file transfer, virtual terminal, and electronic mail. The purpose of this layer is to serve as the window between correspondent application processes which are using the OSI to exchange meaningful data. Examples of **application layer** protocols include SNMP, RLOGIN, TFTP, FTP, MIME, NFS, and FINGER. Layer 6, the presentation layer, is responsible for data formatting, character code conversion, and data encryption of data generated in the **application layer**. This layer is not always implemented in a network protocol. Layer 5, the session layer, provides for negotiation and establishment of a connection with another node. To do this, the session layer provides services to (a) establish a session connection between two presentation entities and (b) support orderly data exchange interactions. This includes establishing, maintaining, and disconnecting a communication

link
between two stations on a network, as well as handling name-to-station address
translation. (This is similar to placing a call to someone on the telephone
network with knowing only his/her name, wherein the name is reduced to a phone
number in order to establish the connection).

Brief Summary Text - BSTX (15):
Layer 2, the data link layer, is responsible for transfer of addressable
units of information, frames, and error checking. This layer synchronizes
transmission and handles frame-level error control and **recovery** so that
information can be transmitted over the physical layer. Frame formatting and
cyclical redundancy checking (CRC), which checks for errors in the whole frame,
are accomplished in this layer. It also provides the physical layer addressing
for transmitted frame. Serial Line IP (SLIP) and point-to-Point Protocol (PPP)
are examples of data link protocols. Finally, layer 1, the physical layer,
handles the transmission of binary data over a communications network. This
layer includes the physical wiring (cabling), the devices that are used to
connect a station's network interface controller to the wiring, the signaling
involved to transmit/receive data, and the ability to detect signaling errors
on the network media. ISO 2110, IEEE 802, and IEEE 802.2 are examples of
physical layer standards.

Claims Text - CLTX (29):
18. The computer system of claim 17, wherein said lack of activity on said
management interface is **detected** by said auto-polling unit **detecting** no **data**
transfers on said serial **data** signal for a **predetermined** number of cycles on
said clock signal.

US-PAT-NO: 5513314

DOCUMENT-IDENTIFIER: US 5513314 A

TITLE: Fault tolerant NFS server system and mirroring protocol

---------- KWIC ---------

Brief Summary Text - BSTX (13):
The known use of NFS shadowing at the server system level relies on delayed writes of shadowed data from a primary to a secondary server system. NFS server level shadowing thus requires only the real-time logging of all data modifications stored on one server to be replicated to at least the second server. The inter-server transfer of such logged data is performed as a low priority background task so as to have minimal impact on the normal function and performance of the primary server system. Even so, the delayed background transfer of logged data from the primary to **backup** server system may consume a substantial portion of the network resources of a primary server. Another problem with NFS server shadowing is that, at the point of any failover, the delayed write of logged data to the surviving system creates an exposure window for the loss of data.

Brief Summary Text - BSTX (19):
Another advantage of the present invention is that failover between a mutually fault tolerance protected server systems of the present invention is relatively instantaneous. That is, the failure detection aspect of the protocol of the present invention can detect and handle failure events consistent with **recovery** from normal NFS error conditions.

Brief Summary Text - BSTX (20):
A further advantage of the present invention is that no additional hardware and minimal additional software is required for the implementation of the present invention. The protocol utilizes the same network connectivity existent for communication with a client to establish the fault tolerance data

communication path between two or more fault tolerance server network systems.
The only specific hardware cost involved is the cost of providing for additional disk data storage on each of the server systems that functions as a
mirror **back-up** to the filesystem storage of another server system. A related
advantage is that no change is required either to the system software or
hardware of a client station in order to make use of the present invention.
Furthermore, administration of the present invention is centralized on the
server systems.

Brief Summary Text - BSTX (22):
A still further advantage of the present invention is that fault tolerant
operation is established in a flexible manner that allows any filesystem or
other data object to be established as either a primary or **back-up** fault
tolerant protected element. Consequently, load sharing between multiple file
servers may be readily established to minimize the practical implications of
establishing fault tolerance behavior between the file servers with respect to
specific fault tolerance protected filesystems or data objects.

Brief Summary Text - BSTX (23):
Yet still another advantage of the present invention is that each primary
file server, succeeding a failover event or other operational data handling
inconsistency, may readily establish a local record of all data modifications
occurring from the point in time of the failover event, thereby permitting a
rapid data reconstruction of the **back-up** file server prior to re-establishing
the fault tolerant pairing of filesystems.

Drawing Description Text - DRTX (6):
FIG. 4 provides a state transition diagram illustrating the sequence of
states executed in the event of a **back-up** NFS server failure to complete a
client write request in accordance with a preferred embodiment of the present
invention;

Detailed Description Text - DETX (4):
In accordance with the preferred embodiments of the present

invention, a fault tolerant protocol is implemented for a specific class of remote procedure calls (RPCs) transferred via the LAN 16 as a series of one or more datagrams. Specifically, the class of RPCs encompassed by the fault tolerant protocol include those known as Network Filesystem (NFS) requests. In general, NFS requests provide for two categories of operations: inquiries and updates. Inquiry requests include read data, get attributes, look up, read directory, read link, status, and null. Update requests include write data, set attributes, rename, remove directory, remove file, link, create file, make directory, and make symbolic link. These NFS requests are **monitored** and managed by the fault tolerant protocol of the present invention in a manner that results in the mirroring of all **data** within **predetermined** filesystems present on a primary 12 and least one secondary 14 file server. The mirroring of data to both the primary and secondary file servers 12, 14 is performed essentially concurrently in response to any client workstation 18, 20 that issues NFS requests with respect to the mirrored filesystems.

Detailed Description Text - DETX (13):
On both the primary files server 12 and secondary file server 14, the datagram representing the NFS write request is processed by a substantially conventional TCP/IP stack. In relevant part, this network stack includes a physical layer, a data link layer, a network layer, a transport layer, a session layer and an. **application layer**.

Detailed Description Text - DETX (16):
Finally, the **application layer** provides for well-known file services, such as file transfer and remote file access. An NFS server layer is the preferred embodiment of the **application layer** used by the present invention. Each read, write or other NFS request is managed through the NFS server under the control of generally respective network control processes (conventionally nfsd processes).

Detailed Description Text - DETX (33):
In each of these events, the primary server 12 is left sleeping on

the DRC
entry for an acknowledgment datagram that is not received. However, in accordance with the present-invention, a sleep timer is set by the primary
server 12 in putting the nfsd process to sleep on DRC entry. The nfsd process
awakes 86 on timeout of the sleep timer in the absence of any received acknowledge datagram. Alternately, the sleep timer is effectively expired upon
the aging of the DRC entry through operation of the DRC-LRU algorithm. In
either event, the primary server 12 then transitions to a **backup** failure
recovery mode 88.

Detailed Description Text - DETX (34):
Where the **backup** failure occurs in a circumstance where the mirrored filesystems of the active group continue to be properly available, and the
integrity of the virtual server is intact but for the failure to receive the
acknowledgment datagram, a partial resynchronization of the mirrored file
systems is possible. The availability of mirrored filesystems is established
by use of a heart-beat protocol, preferably performed on a per mirrored filesystem basis by all active group servers on the LAN 16, to continually
broadcast evidence of the continuing availability of the corresponding exported
filesystems on the LAN 16. Preferably, this heart-beat protocol is implemented
through the issuance of a custom UDP datagram multicast to the fileservers of
the active group. Where such heart-beat datagrams are still being exchanged
between at least the active group servers 12, 14 of a given mirrored filesystem, thereby indicating that the mirror filesystem on the secondary
server 14 is available even though a sleep event for an acknowledge packet has
timed out on the primary server 12, the primary server 12 may intentionally
withhold issuing any completion datagram to the client 26. Subject to the
conventional operation of the NFS protocol, the client 26 will ultimately
time-out waiting for the completion datagram and reissue the NFS write request.

Detailed Description Text - DETX (57):
Another consideration dealt with by the present invention is the **recovery** of
mirrored status between mirror filesystems on the primary and secondary servers
12, 14 following the correction of the cause of a failover event. A

number of
different approaches to **recovery** are contemplated by the present invention.
The simplest **recovery** technique is to simply quiesce the failover surviving
server and copy all data files within the surviving mirror filesystem to the
mirror filesystem of the rejoining server. This copy can be performed selectively based at least on last modification timestamps that are subsequent
to the moment of the failover event. The exact time of occurrence of the
failover event is preferably recorded at the time of occurrence by the surviving server through the /etc/syslogd service or an equivalent event
logging service.

Detailed Description Text - DETX (58):
The preferred partial resynchronization approach to **recovery** is to provide
for the logging of all file data modifications, including file creations and
deletions, that are made to the mirror filesystem of the surviving server from
the point of the failover event to the quiescing of read/write activity in
preparation for **recovery**. The **backup** logs may store the accumulated incremental changes to the data files present on the mirror filesystem. While
the mirror filesystems are otherwise quiesced, this log may simply be transferred or replayed from the surviving server to the rejoining server,
thereby resynchronizing the data present on the mirrored filesystems. Where
write activity is not quiesced on the surviving server during the partial
resynchronization, a new log of write information can be accumulated by the
surviving server while writing a prior log to the rejoining server. Successive
logs will be smaller until either no writes occur to the affected filesystem
during the transfer of a log or, once the current log is of a sufficiently
small size, writes to the filesystem on the surviving fileserver are temporarily suspended or held off until the final log is written to the rejoining server.

Detailed Description Paragraph Table - DETL (1):
TABLE I

Duplicate Request Cache
struct dupreq [u.sub.-- long dr.sub.-- xid; /* transaction ID */ u.sub.-- long dr.sub.-- proc; /* procedure called */ u.sub.-- long dr.sub.-- vers; /* version number called

```
*/  u.sub.-- long  d.sub.-- prog;  /* program number  called */  char
dr.sub.--
inprogress;  /* 1 if request is  in progress */  char dr.sub.-- status;
 /*
status of  original reply */  u.sub.-- short  dr.sub.-- port;  /* UDP
port of
sender */  struct in.sub.-- addr  dr.sub.-- hostaddr;  /* IP address of
 sender
*/  struct timeval  dr.sub.-- timestamp;  /* time stamp */  struct
duprply
*dr.sub.-- reply;  /* reply for non-  idempotent req */  struct dupreq
*dr.sub.-- next;  /* LRU cache  chain */  struct dupreq  *dr.sub.--
chain;  /*
hash chain */  #ifdef FTNFS  char dr.sub.-- ack;  /* bit mask of
backup acks
rec'd */  u.sub.-- long  dr.sub.-- inode;  /* new file inode  and
generation */
u.sub.-- long  dr.sub.-- generation;  /* as provided by  the primary */
 #endif
/* FTNFS */  ];
```

US-PAT-NO: 6393568

DOCUMENT-IDENTIFIER: US 6393568 B1

TITLE: Encryption and decryption system and method with content
analysis provision

---------- KWIC ---------

Detailed Description Text - DETX (5):
Preferably, the system decrypts and runs virus detection on each document or
file as the file is initially received by the computer or prior to transferring
of the data for use by a target application Such a real time process can
prevent a virus from being unknowingly unleashed as the file is first
encountered by the system, as compared to conventional systems that would have
otherwise allowed the virus to go undetected as an encrypted document. Also,
if desired the combined decryption and content analysis, such as virus
detection, can be run as a batch analysis as part of a maintenance program to
decrypt all files in a hard drive or network server on a pre-determined
schedule to check for viruses in decrypted documents. If desired, the content
analysis application can be run on a file server which contains a **backup** copy
of data. Significant results of any content analysis can then be summarized
and conveyed for use on the original files, with the advantage that performance
penalties of the overall analysis are minimized for online systems containing
the original files.

Detailed Description Text - DETX (8):
The content inspection application 18 receives decrypted data from memory as
decrypted and stored by the cryptographic application 10. The content
inspection application 18, such as a virus **detection** program, analyzes the
decrypted **data** to determine whether or not **predefined** content is contained in
the decrypted **data,** or to determine what further action or processing should be
applied to the **data** under inspection. For example where the packet content
inspector 18 is a virus detection application, the cryptographic application 10

launches the virus detection application after decryption of the data or a portion of the data has been completed. The virus detection application then evaluates the decrypted data to determine, for example, whether or not an infection is present in the data and generates inspection status data 20, such as infection status data. Consequently, unlike conventional cryptographic systems, virus detection is launched by the cryptographic application and performed on decrypted data so that latent **viruses are not present in the stored data**.

Detailed Description Text - DETX (10):

The cryptographic application assesses whether the user has access to the decryption keys or whether necessary decryption keys need to be obtained form another source, as indicated in block 34. For example, where the batch of data to be decrypted and analyzed is from many different users with varying encryption keys, the cryptographic system may have to obtain additional decryption key information from another source. If the computer performing the decryption has access to the decryption keys, the cryptographic application decrypts the data or file as shown in block 36. The cryptographic application then sends a content inspection request, such as a virus detection request, to launch the virus detection application as indicated in block 38. The virus detection application 18 then analyzes the decrypted data to determine whether a **virus is present within the data** as indicated in block 40 and 42.

Detailed Description Text - DETX (22):

If desired, the content analysis application can be run on a file server which contains a **backup** copy of data. A batch decryption and content analysis operation is performed on all of the **back up** copies. Significant results of any content analysis can then be summarized and conveyed for use on the original files by the server, with the advantage that performance penalties of the overall analysis are minimized for online systems containing the original files. Hence the server generates content analysis status information and sends the information to an appropriate node in the network designated as a user, owner or administrator of the file or packet.

US-PAT-NO: 5623600

DOCUMENT-IDENTIFIER: US 5623600 A

TITLE: Virus detection and removal apparatus for computer networks.

---------- KWIC ----------

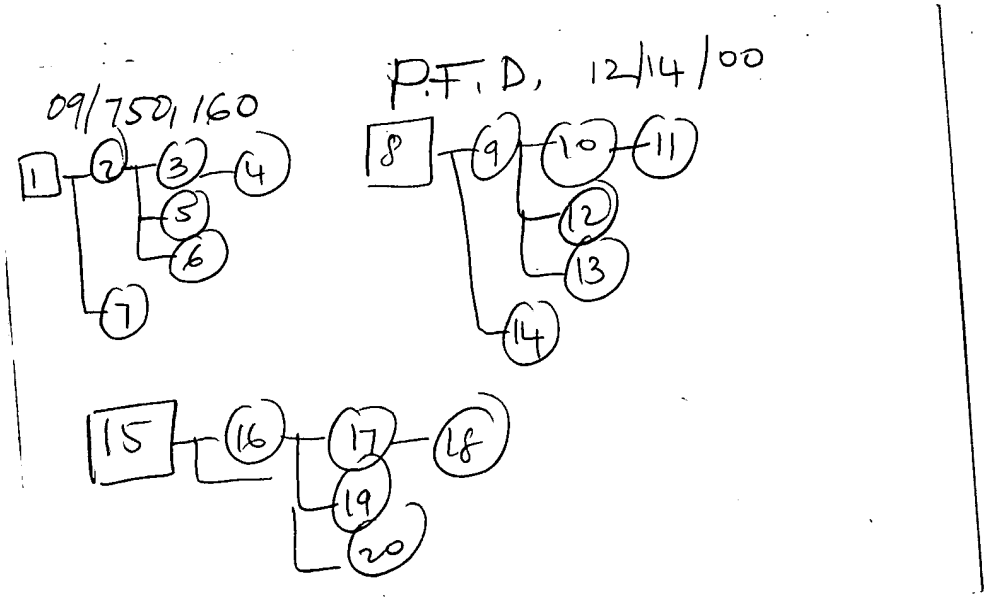
Brief Summary Text - BSTX (12):
The present invention also comprises a method for processing a file before transmission into the network and a method for processing a file before transmission from the network. The preferred method for processing a file comprises the steps of: receiving the **data** transfer command and file name; transferring the file to the proxy server; performing virus **detection** on the file; determining whether the file contains any viruses; transferring the file from the proxy server to a recipient node if the file does not contain a virus; and performing a **preset** action with the file if it does contain a virus. The present invention also includes methods for processing messages before transmission to or from the network that operate in a similar manner.

Detailed Description Text - DETX (8):
While the apparatus of the present invention, in particular the FTP proxy server 60 and SMTP proxy server 62, has been described above as being located and preferably is located on the gateway node 33, those skilled in the art will realize that the apparatus of the present invention could also be included on a FTP server or a world wide web server for scanning files and **messages as they are downloaded** from the web. Furthermore, in an alternate embodiment, the apparatus of the present invention may be included in each node of a network for performing virus detection on all messages received or transmitted from that node.

Detailed Description Text - DETX (9):
As best shown in FIG. 4, the CPU 42 also utilizes a protocol layer hierarchy

to communicate over the network. The protocol layers of the hierarchy of the present invention are shown in FIG. 4 in comparison to the ISO-OSI reference model, for example. The protocol layers 410-426 of the hierarchy of the present invention are similar to the prior art protocol layers for the lower four layers 400-403 including: (1) a physical layer 400 formed of the transmission media 410; (2) a data link layer 401 formed of the network interface cards 411; (3) a network layer 402 formed of address resolution 412, Internet protocol 413 and Internet control message protocol 414; and (4) a transport layer 403 formed of the transmission control protocol 415 and a user datagram protocol 416. Corresponding to the presentation 405 and session 404 layers, the protocol hierarchy of the present invention provides four methods of communication: a file transfer protocol 417, a simple mail transfer protocol 419, a TELNET protocol 419 and a simple network management protocol 420. There are corresponding components on the **application layer** 406 to handle file transfer 423, electronic mail 424, terminal emulation 425, and network management 426. The present invention advantageously detects, controls and eliminates viruses by providing an additional layer between the **application layer** 406 and the presentation layer 405 for the gateway nodes 33. In particular, according to the hierarchy of the present invention, a FTP proxy server layer 421 and a SMTP proxy server layer 422 are provided. These layers 421,422 operate in conjunction with the file transfer layer 423 and file transfer protocol 417, and the electronic mail layer 424 and the SMTP protocol layer 418, to process file transfers and messages, respectively. For example, any file transfer requests are generated by the file transfer application 423, first processed by the FTP proxy server layer 421, then processed by the file transfer protocol 417 and other lower layers 415, 413, 411 until the data transfer is actually applied to the transmission media 410. Similarly, any messaging requests are first processed by the SMTP proxy server layer 418, and thereafter processed by the SMTP protocol and other lower layers 415, 413, 411 until the physical layer is reached. The present invention is particularly advantageous because all virus screening is performed below the application

level. Therefore, the applications are unaware that such virus detection and elimination is being performed, and these operations are completely transparent to the operation of the application level layers 406. While the FTP proxy server layer 421 and the SMTP proxy server layer 422 have been shown in FIG. 4 as being their own layer to demonstrate the coupling effects they provide between the file transfer layer 423 and file transfer protocol 417, and the electronic mail layer 424 and the SMTP protocol layer 418, those skilled in the art will realize that the FTP proxy server layer 421 and the SMTP proxy server layer 422 can also be correctly viewed as being part of the file transfer protocol layer 417 and the SMTP protocol layer 418, respectively, because they are invisible or transparent to the **application layer** 406.



US-PAT-NO: 5889943

DOCUMENT-IDENTIFIER: US 5889943 A

TITLE: Apparatus and method for electronic mail virus detection and elimination

---------- KWIC ---------

Abstract Text - ABTX (1):
The detection and elimination of viruses on a computer network is disclosed. An apparatus for detecting and eliminating viruses which may be introduced by messages sent through a postal node of a network electronic mail system includes polling and retrieval modules in communication with the postal node to determine the presence of unscanned **messages and to download** data associated with them to a node for treatment by a virus analysis and treatment module. A method for detecting and eliminating viruses introduced by an electronic mail system includes polling the postal node for unscanned **messages, downloading** the messages into a memory of a node, and performing virus detection and analysis at the node.

Brief Summary Text - BSTX (16):
The present invention also comprises a method for processing a file before transmission into the network and a method for processing a file before transmission from the network. The preferred method for processing a file comprises the steps of: receiving the **data** transfer command and file name; transferring the file to the proxy server; performing virus **detection** on the file; determining whether the file contains any viruses; transferring the file from the proxy server to a recipient node if the file does not contain a virus; and performing a **preset** action with the file if it does contain a virus. The present invention also includes methods for processing messages before transmission to or from the network that operate in a similar manner.

Brief Summary Text - BSTX (18):
The present invention also comprises a method for detecting and

eliminating
viruses which may spread throughout a network in messages accessed by an
electronic mail system. Preferably, the postal node is polled from the client
node for unread messages, unread **messages are downloaded** into the memory of a
client node, the messages are scanned for the presence of viruses, and
corrective action taken.

Detailed Description Text - DETX (8):
While the apparatus of the present invention, in particular the FTP proxy
server 60 and SMTP proxy server 62, has been described above as being located
and preferably is located on the gateway node 33, those skilled in the art will
realize that the apparatus of the present invention could also be included on a
FTP server or a world wide web server for scanning files and **messages as they
are downloaded** from the web. Furthermore, in an alternate embodiment, the
apparatus of the present invention may be included in each node of a network
for performing virus detection on all messages received or transmitted from
that node.

Detailed Description Text - DETX (9):
As best shown in FIG. 4, the CPU 42 also utilizes a protocol layer hierarchy
to communicate over the network. The protocol layers of the hierarchy of the
present invention are shown in FIG. 4 in comparison to the ISO-OSI reference
model, for example. The protocol layers 410-426 of the hierarchy of the
present invention are similar to the prior art protocol layers for the lower
four layers 400-403 including: (1) a physical layer 400 formed of the
transmission media 410; (2) a data link layer 401 formed of the network
interface cards 411; (3) a network layer 402 formed of address resolution 412,
Internet protocol 413 and Internet control message protocol 414; and (4) a
transport layer 403 formed of the transmission control protocol 415 and a user
datagram protocol 416. Corresponding to the presentation 405 and session 404
layers, the protocol hierarchy of the present invention provides four methods
of communication: a file transfer protocol 417, a simple mail transfer protocol
419, a TELNET protocol 419 and a simple network management protocol 420. There

are corresponding components on the **application layer** 406 to handle file transfer 423, electronic mail 424, terminal emulation 425, and network management 426. The present invention advantageously detects, controls and eliminates viruses by providing an additional layer between the **application layer** 406 and the presentation layer 405 for the gateway nodes 33. In particular, according to the hierarchy of the present invention, a FTP proxy server layer 421 and a SMTP proxy server layer 422 are provided. These layers 421, 422 operate in conjunction with the file transfer layer 423 and file transfer protocol 417, and the electronic mail layer 424 and the SMTP protocol layer 418, to process file transfers and messages, respectively. For example, any file transfer requests are generated by the file transfer application 423, first processed by the FTP proxy server layer 421, then processed by the file transfer protocol 417 and other lower layers 415, 413, 411 until the data transfer is actually applied to the transmission media 410. Similarly, any messaging requests are first processed by the SMTP proxy server layer 418, and thereafter processed by the SMTP protocol and other lower layers 415, 413, 411 until the physical layer is reached. The present invention is particularly advantageous because all virus screening is performed below the application level. Therefore, the applications are unaware that such virus detection and elimination is being performed, and these operations are completely transparent to the operation of the application level layers 406. While the FTP proxy server layer 421 and the SMTP proxy server layer 422 have been shown in FIG. 4 as being their own layer to demonstrate the coupling effects they provide between the file transfer layer 423 and file transfer protocol 417, and the electronic mail layer 424 and the SMTP protocol layer 418, those skilled in the art will realize that the FTP proxy server layer 421 and the SMTP proxy server layer 422 can also be correctly viewed as being part of the file transfer protocol layer 417 and the SMTP protocol layer 418, respectively, because they are invisible or transparent to the **application layer** 406.

Detailed Description Text - DETX (39):

Preferably, the mail polling module 282 includes routines for polling or accessing the postal node 232 to determine whether any new messages have arrived for the client and remained unscanned. Such routines are arranged to communicate with the mail management 292 and storage areas 264 of the postal node 232 and preferably emulate the polling routines of the electronic mail program 274, 292 used by the network 200. The polling routines include conventional ones, and may, for example, implement the Vendor Independent Messaging (VIM) interface of the electronic mail system or the Dynamic Data Exchange (DDE) interface. The polling routines may literally emulate the routines used by the electronic mail program 274, 292 and may be set according to the configuration settings of the mail scanning manager 280. The polling routines are preferably executed on a fixed time interval such as every 30 seconds to poll the postal node 232 and determine whether any unscanned message addressed to the client node 230 has been received. The polling module 282 uses and maintains the date in the scanned message FIFO buffer 285. The scanned message FIFO buffer 285 is a table that list the messages at the post office which are addressed to the client node, are unread and have already been scanned for viruses. As illustrated in FIG. 11d, the scanned message FIFO buffer 285 is preferably a portion of memory 248 fixed in sized to hold a plurality of entries, each entry having a message identification number, header information and one or more status bits. The buffer 285 is preferably a circular buffer or FIFO buffer, in that, once the buffer is filled with information on scanned messages, the oldest entry in the FIFO will be deleted to make room for the next entry. The process of polling for unscanned messages is performed by using conventional routines to determine if there is an unread message addressed to the client node 230 at the postal node 232. If there is, the polling retrieves the unique identification number (and other header information if necessary) and compares the unique identification number to the unique identification numbers stored in the scanned message FIFO buffer 285. If the unique identification number for the unread message is in the scanned

message FIFO buffer 285, then the **message is not download** to the data buffer,
the polling continues with the next unread message at the postal node. However, if the unique identification number for the unread message is not in
the scanned message FIFO buffer 285, then the unique identification number is
passed to the retrieval module 283 so that the message and its contents can be
download to the data buffer 284.

Detailed Description Text - DETX (45):

Now referring to FIG. 12, a preferred method of operation 1200 for the
electronic mail scanning apparatus is shown. Preferably, the postal node 232
is polled 1205 by emulating the polling routines of the electronic mail system
to determine whether any unscanned messages that are addressed to a predetermined recipient are present. When unscanned **messages for the predetermined** recipient are **detected** at the postal node 232, the mail scanning
apparatus downloads 1210 the **message,** including any attachments, to memory 248
of the client node 230 assigned to the **predetermined** recipient. The preferred
method then scans 1215 the message and attachment stored in memory 246 to
determine 1220 whether the message or attachment contains a virus. Then in
step 1220, the method determines whether the message includes a virus. If the
message is found to have a virus, the mail scanning apparatus may then take
corrective action 1225 regarding the infected message, by either removing the
virus, sending a warning as part of the message, deleting the message or
forwarding the message to a system administrator. Preferably, the polling
routines 1205 operate without user input and without activation of the local
electronic mail program 274 at the client node 230 to allow for unobtrusive
detection and operation in the background.

Claims Text - CLTX (16):

11. The apparatus of claim 9, wherein the virus treatment module includes
routines for performing a **preset** action on the **message** when a virus is **detected**
in the message.

Claims Text - CLTX (19):

14. The apparatus of claim 3, wherein the message is intended for

access by
a first node, the message detecting module and the virus treatment module
reside at a server; and the virus treatment module includes routines for
performing a **preset** action on the **message** when a virus is **detected in the
message**.

Claims Text - CLTX (37):
27. The method of claim 25, wherein the step of treating the **message**
comprises performing a **preset** action on the **message** when a virus is **detected in
the message**.

Claims Text - CLTX (40):
30. The method of claim 19, wherein the message is intended for access by a
first node, the step of detecting the presence of a message is undertaken at a
server; and the step of treating the **message** comprises performing a **preset**
action on the **message** when a virus is **detected in the message**.